Surface Reconstruction Using Oriented Charges

Jianning Wang* Manuel M. Oliveira† Hui Xie* Arie E. Kaufman*
* Center of Visual Computing, Computer Science, Stony Brook University
† Instituto de Informática - UFRGS - Brazil

ABSTRACT

We introduce the notion of oriented charges to compute distance fields from unorganized point clouds. Unlike traditional implicit function methods that usually require normal information, this new approach only uses information about the positions of the samples. Our approach is adaptive, using an octree to reorganize the input points, thus avoiding difficulties often associated with the use of local distance fields. The resulting representation is simple, efficient and general in the sense that it handles complex models of arbitrary topology, is robust to the presence of noise and fills holes automatically.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Virtual reality

Keywords: surface reconstruction, oriented charges, distance fields, implicit representation

1 INTRODUCTION

In recent years, there has been a growing interest in the construction of virtual replicas of real objects. Such an interest has been driven by applications such as virtual museums, entertainment, and analysis of forensic records, among others. However, building virtual replicas of real objects and scenes is a complex task for which the use of conventional modeling techniques is not appropriated. The use of 3D laser scanning is becoming an increasingly popular alternative to conventional modeling [15] and large amounts of data in the form of samples of geometry are available [7].

The problem of surface reconstruction from unorganized point clouds is challenging and ill-posed. Moreover, in range scans acquired from real objects, noise tends to contaminate the data, making the reconstruction task harder. Whenever it is difficult to accurately estimate normals at the samples (*e.g.*, in noisy datasets), existing implicit methods tend to fail to create distance fields that correctly represent surface details and topology.

In this paper, we introduce a new implicit method that only requires information about the positions of points and, therefore, can be used for surface reconstruction from both clean and noisy datasets. Our method is based on two key observations: (1) a highly accurate distance field is not necessary for reconstruction when the data is quite noisy. In these cases, it is sufficient to have an approximate smooth distance field whose error near the surface is bounded by the amount of noise in the data. (2) Assuming the surface to be reconstructed is closed, one can use a bounding box containing the point cloud to orient the surface (*i.e.*, to define its exterior). Based on these observations, we approximate the distance field using an adaptive and hierarchical framework. Although no formal proofs are presented, we provide some intuitive evidence that the proposed approach guarantees the orientability of the reconstructed surfaces, preserves topology, and that the reconstruction error is bounded by the amount of noise and by the sample density of the input point cloud.

Figure 1: Reconstruction of David's head using our algorithm.

The contributions of this paper include:

- An improved implicit method for surface reconstruction from point clouds that only requires information about the positions of the samples and is robust to the presence of noise and missing data (Sections 3 and 4).
- A new algorithm for computing approximate distance fields based on oriented charges (Section 3);
- A hierarchical and adaptive representation for performing surface reconstruction at different levels of detail as well as hole filling (Section 7).

2 PREVIOUS AND RELATED WORK

The problem of reconstructing a continuous function from unorganized datasets in R^n has been extensively studied over the past decades [10, 11, 12]. In recent years, the problem of surface reconstruction in 3D has attracted the attention of several researchers [2, 6, 14, 17, 23, 25].

Computational geometry methods use mechanisms such as Delaunay triangulation [2, 9] and region growing [3]. Such methods interpolate the original points, being sensitive to the presence of noise. By fitting smooth functions to the input points, algebraic methods [21] handle noisy datasets, but cannot deal with arbitrary topology.

Implicit function methods are the most popular approach for surface reconstruction from point clouds [6, 12, 14, 17, 23]. Essentially, these methods are based on the use of a sign distance function and can naturally handle different topologies. They require a way

*email:{jianning,xhui,ari}@cs.sunysb.edu
†email:oliveira@inf.ufrgs.br

to distinguish between the inside and outside of (closed) surfaces, which is usually achieved by using information of surface normals at the samples.

Turk and O'Brien [23] and Carr et al. [6] build signed distance functions using non-compact radial basis functions (RBF). Morse et al. [14] use compactly-supported RBFs based on the work by Wendland [24] to achieve local control and improved memory and computational time requirements. Ohtake et al. [17] uses a method based on the partition of the unity to achieve similar results. Dinh et al. [8] use a smooth term in the radial basis function to handle noisy in sparse range datasets obtained from passive vision techniques. Dinh's approach requires solving a linear system, which constrains the size of datasets. Carr et al. [5] handle surface reconstruction from noisy datasets by fitting an RBF and then changing the basic function during the evaluation. As in [6], the method requires normals for defining the interior and exterior of the surfaces.

The idea of oriented charges was inspired by the notion of *oriented particle systems* [20], but it distinguishes itself from the latter, which has no relation to distance fields. In our approach, a global distance field is formed by blending local distance fields of nearby oriented charges. Our approach also differs from Ohtake's algorithm [17] because we compute the implicit function based only on the location of points while Othake's approach also uses information about normals. Moreover, our approach is less sensitive to noise, producing reasonable results even for very noisy datasets.

3 Oriented Charges

An *oriented charge* $Q_i(p_i, q_i, n_i)$ (OC) is a charge instantiated at a point p_i, with magnitude q_i and orientation n_i, where $\|n_i\| = 1$. It implies a linear local distance field at position x given by

$$F_i(x) = q_i - (x - p_i) \cdot n_i \tag{1}$$

Let O be an octree defining a subdivision of the space containing a point cloud. The adaptive subdivision of the octree is trigged by the number of points inside a given node, up to a maximum specified octree level. In our current implementation, we recursively subdivide an octree node until it contains no more than six samples. In order to approximate the distance field induced by the surface represented by the point cloud, a set of oriented charges are instantiated at the neighborhood of the nodes containing the original samples (Section 4).

Each charge Q_i has an associated linear distance field, whose iso-surfaces are the set of parallel planes perpendicular to n_i. This can be easily seen by rewriting Equation 1 as $F_i(x) = q_i - \alpha$, where $\alpha = (x - p_i) \cdot n_i$. Although, according to Equation 1, F_i can assume arbitrarily large values both in the positive and negative ranges, the local distance field of an oriented charge is made effective only within a $3t \times 3t \times 3t$ neighborhood centered at the oriented charge itself. Here, t is the side of the associated octree node. To combine overlapping distance fields into a global one, we use a scaled version of a Gaussian function as the weight for each oriented charge Q_i

$$\omega_i(x) = 2^l e^{-\frac{\|p_i - x\|^2}{\sigma_l^2}} \tag{2}$$

where l is the level of its associated octree node and σ_l varies with the size of that octree node (in our experiments we used $\sigma_l = 1.5t$). This weight function was designed such that oriented charges at coarser levels have larger areas of influence than at finer levels. The field induced by multiple oriented charges is a weighted average of Equation 1:

$$F(x) = \frac{\sum_i F_i(x) * \omega_i(x)}{\sum_i \omega_i(x)} \tag{3}$$

3.1 Octree Nodes, Crusts and Frontiers

Given an octree O enclosing a set of samples in 3D, its nodes can be classified as either *exterior*, *interior*, *boundary*, *hole* or *unknown* nodes. The notion of octree-node inside-outside classification was first introduced by Samet [18] and has also been recently used to support CSG operations in point-based graphics [1].

Initially, each octree node is labeled either as a boundary or as an unknown node, depending on whether it contains some or no samples, respectively. After that, all nodes of the octree are traversed and once an unknown node is found, it is re-labeled as either exterior, interior or hole node. Exterior/interior nodes are outside/inside the object (*i.e.*, outside/inside the object's closed surface representation). Boundary nodes and hole nodes intersect the zero-set surface of the object. We use the term *crust* to refer to a closed thin shell of octree nodes, all at the same level of the octree. The *surface crust* is composed only of boundary and hole nodes. The *exterior* and *interior crusts*, on the other hand, only have exterior and interior nodes, respectively, and together tightly bound the surface crust. *Oriented charges are only instantiated in the exterior and interior crusts*.

4 Surface Representation with OCs

In order to identify the exterior and interior crusts where oriented charges will be instantiated, we first find two crusts which are absolutely outside and inside the surface and refer to them as the *exterior* and *interior frontiers*, respectively. The propagation of frontiers is based on an adaptive refinement approach that re-labels unknown nodes and finally finds the surface, the exterior and the interior crusts (Figure 2 and Section 4.2).

The algorithm for instantiating oriented charges is shown in Algorithm 1. Starting from a coarser to a finer level of the octree, first the exterior and interior frontiers are identified (Section 4.1). These frontiers are propagated to compute the crusts at that given level of the octree (Section 4.3). The resulting information is then used as input to the subsequent finer level. Once we reach the finest level, oriented charges are instantiated at the exterior and interior crusts.

for *octree_level= 1 to MAX* **do**
    Find the exterior and interior frontiers for the current level;
    Propagate the frontiers to find the exterior and interior crusts;
end
Instantiate OCs in the exterior and interior crusts;

Algorithm 1: Instantiation of oriented charges

4.1 Finding and Propagating Frontiers

We revise the tagging algorithm of Zhao et al. [25] to find the exterior and interior surface crusts. We make the original algorithm adaptive to sampling density and progressively enlarge the trusted exterior/interior regions. The algorithm assumes that the outer nodes of the octree contain no samples and, therefore, can be safely set as exterior nodes (Figure 2(a)). Starting at a coarser level, the exterior frontier is propagated until nodes containing at least one sample are reached. These nodes are labeled as boundary nodes. The remaining empty nodes (*i.e.*, that could not be reached from the exterior frontier) are then labeled as interior nodes (Figures 2(b)). At this point, the algorithm refines this initial tagging (Figures 2 (c) to (f)).

During a node refinement, the children of an exterior/interior/hole node are also labeled as exterior/interior/hole

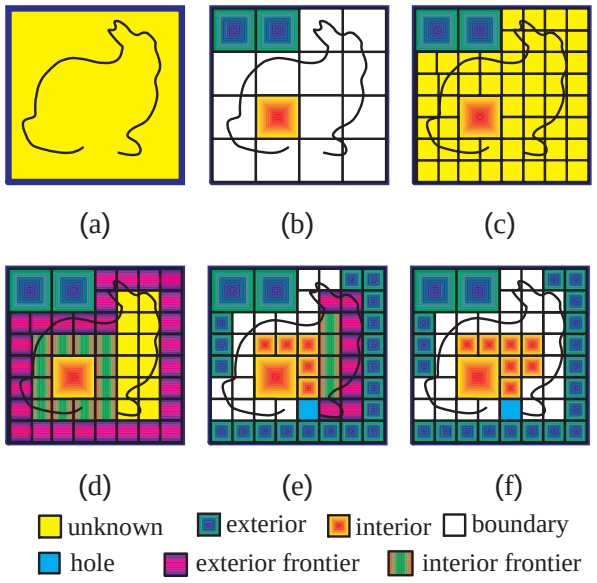

Figure 2: (a) Initial exterior frontier. (b) Exterior and surface crusts at the coarsest one. Remaining unknown nodes set as inside. (c) Boundary nodes are subdivided and labeled as unknown. (d) New exterior/interior frontiers. (f) Final classification.

nodes. The children of a boundary node, however, can become exterior, interior, boundary or hole nodes. Thus, when all the nodes at a coarser level have been labeled, at the next level the algorithm only needs to classify the children of boundary nodes. Initially, we mark all children of boundary nodes as unknown nodes (Figure 2(c)). The refinement proceeds using the children of the of previous-level exterior/interior frontier as the new exterior/interior frontier (Figure 2(d)). The final result is illustrated in Figure 2(f).

4.2 The Crusts

The next step is to find the interior, exterior and surface crusts. First, we compute the average location of samples inside each boundary node. Then, we compute the Euclidean distances between the centers of exterior/interior frontier nodes and the average locations of the samples in the boundary nodes under their radius of influence (a $3 \times 3 \times 3$ neighborhood). We record the minimal distance in each frontier node. Based on their minimal distances, nodes in the frontiers are heap sorted. For each iteration, the node at the top of the heap (*i.e.*, with the biggest distance) is taken out and processed as follows. If this node has the smallest distance compared to its adjacent unknown nodes, it will be marked as a hole node. Otherwise, it is discarded and all its adjacent unknown nodes are inserted into the heap. When the iteration ends, boundary and hole nodes compose the surface crust and exterior/interior nodes touching the boundary/hole nodes become the exterior/interior crust (Figure 2(e) and (f)).

4.3 Instantiating Oriented Charges

One oriented charge Q_i is instantiated at the centers of each node in the exterior and the interior crusts. For a given node N_i in the exterior/interior crust, the final magnitude, q_i, and orientation, n_i, of Q_i is defined by the set S of surface-crust nodes in the $3 \times 3 \times 3$ neighborhood of N_i. First, for each node s_k in S, we compute a candidate oriented charge cQ_j for N_i. The orientation of cQ_j is computed as the centerline of the cone with apex at p_i (the center

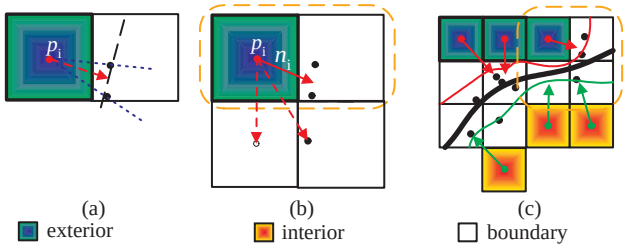

Figure 3: (a) A candidate oriented charge; (b) The candidate with smallest magnitude is selected; (c) The global distance field (the thick black curve) is created combining the distance fields of the exterior and interior crusts.

of N_i) that bounds all samples in s_k. This is illustrated in Figure 4 (a). The magnitude cq_j of cQ_j is computed as

$$\min_{cq_j} |cq_j| \tag{4}$$

such that $(cq_j - (p_i - d_k)) \cdot n_j \geq 0$ holds for at least γ% of the d_k samples inside s_k. γ is a parameter to counteract the effect of noise and outliers. Intuitively, Equation 4 finds, for each external and internal crust node, the closest plane perpendicular to n_j that separates the samples inside s_k into two groups, such that at least γ% of the points lie on one side of the plane. We obtain such a plane using linear programming. Then, the tuple (p_i, cq_i, n_i) with the smallest $|cq_i|$ is selected as the oriented charge Q_i for N_i. The sign of cq_i is set to "+" or "-", depending on whether the node is in the exterior or in the interior crust, respectively. If a node s_k in S is a hole node (*i.e.*, it contains no samples), a virtual sample is instantiated at the center of s_k before computing cq_i and n_i using the procedure just described. This situation is illustrated in Figure 4(b), where a virtual sample has been instantiated at the center of the bottom left surface crust.

To create oriented charges at a coarser level of the octree, the oriented charges in adjacent nodes are considered. Suppose we have all oriented charges at a finer level. For every such oriented charge Q_i, a virtual point is computed as located at $p_i - q_i n_i$. To compute oriented charges at the coarser level, we collect virtual points instead of real points within its neighborhood before Equation 4 is applied. This bottom-up process computes oriented charges from the finest level to the coarsest level.

5 OC Properties

This section discusses some of the properties of OCs, providing some intuitive evidence about them.

Surface Orientability Preservation. Since the exterior/interior crust bounds the surface of the object tightly, it represents a manifold itself. Besides, all oriented charges in the exterior/interior crust are pointing to the object's surface inwardly/outwardly. Thus, the exterior/interior distance field has correct orientability. The associated gradient field is nonzero everywhere in the surface crust (the internal and external distance fields have opposite signs: negative and positive, respectively) having a direction from outside to inside. When weighted averaging the exterior and interior distance fields, the resulting gradient field is nonzero everywhere on the surface crust. Therefore, inside the region of interest, the resulting distance field has no twists or singular points, and the reconstructed surface is well oriented.

Topology Preservation. Given that the original surface has been sampled at the appropriate sampling rate, the topology of the surface is preserved. The thickness of the thin volume composed by

the exterior, interior and surface crusts decreases as the octree nodes are subdivided. When the thickness is small enough, the manifold represented by the thin volume has the same topology as the surface. Since the reconstructed surface lies within the thin volume, it will approximate the surface accurately (*e.g.*, showing the correct connection and genus). The assumption that surfaces are sampled at appropriate rates guarantees correct topology for clean datasets. In the case of noisy ones, we are limited by the amount of noise relatively to the sizes of the local structures of the objects. Thus, small (relative to the amount of noise) structures may not be correctly reconstructed. This situation is illustrated in Figure 8(d), where the small holes under buddha's arms have collapsed due to the presence of large amounts of noise.

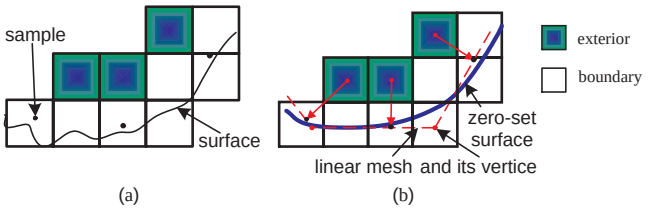

Figure 4: (a) The actual surface and samples. (b) The reconstructed surface.

Error Bounding. A given oriented charge induces a local linear distance field, whose zero-set surface is an infinite plane. When weighted averaging the influence of two adjacent oriented charges, the intersection line of two zero-set planes is part of the resulting zero-set surface. To estimate the error bound, the zero-set surfaces of the exterior and interior distance fields are approximated by piecewise meshes (shown in 2D in Figure 3). Let τ be the error bound in the input samples. Because the influential region of oriented charge is limited within a $3\times3\times3$ subvolume, the error bound on the reconstructed surface is $\tau+\frac{3\sqrt{3}}{2}t$ (*i.e.*, τ plus half of the diagonal of the $3\times3\times3$ subvolume), where t is the size of the octree's leaf nodes.

6 Iso-Surface Extraction and Cost Analysis

Iso-surface extraction is obtained using a variant of the Marching Cubes algorithm [4]. To efficiently improve the accuracy, the size of the marching cubes is made compatible to the smallest node of the octree. Since the details of the surfaces are mainly captured by the oriented charges at the octree's leaf nodes, we use all oriented charges at neighboring nodes at the same and direct coarser levels to compute the corresponding distance fields using Equation 3. While the traditional Marching Cubes algorithm tends to unnecessarily reconstruct many small triangles over flat areas, an adaptive solution could be used to avoid this problem [19].

6.1 Mesh Refinement

Since oriented charges define piecewise linear implicit functions, they may lead to coarse reconstruction in regions with high curvatures. Jeong et al. [13] use subdivision surfaces with adaptive sampling to improve a coarse model. We, instead, address this problem with a mesh-refinement step that follows surface extraction. Since the reconstructed model is already quite close to the real surface, one can move vertices of the mesh along their normals to make them closer to their nearest samples (Figure 5). Notice these normals are computed at the vertices, after mesh extraction and are not used for computing the distance field. The range of adjustment is limited by the size of the finest octree node to avoid mesh self-intersection.

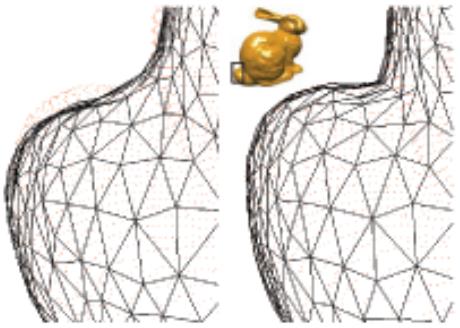

Figure 5: (a) Extracted mesh using Marching Cubes. Small dots are original samples. (b) Final mesh after vertex displacement.

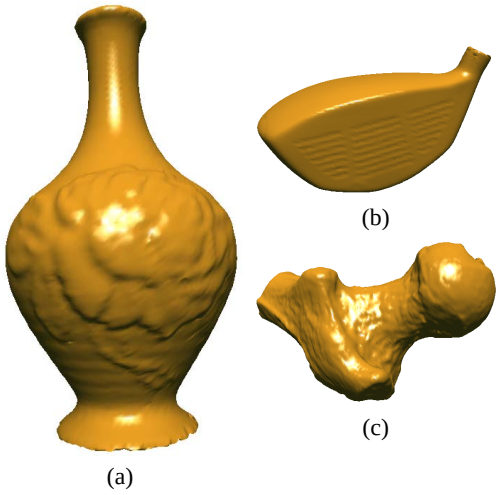

Figure 6: A vase (a), a golf club (b) and a balljoint (c) reconstructed with our algorithm.

6.2 Cost Analysis

Let n be the number of samples in a given point cloud and assume that each octree leaf node contains at most one sample. Since each octree node, except for the first level, must have a parent node, the total number of occupied cells that need to be stored can be expressed as $\Sigma = n+n/8+n/64+\ldots+1$. Σ converges to $(8/7)n$. In our current implementation, for each occupied node we currently store links to its 26 neighbors, making the total memory requirements of the algorithm $27(8/7)n \approx 31n = O(n)$. The cost is linear on the number of samples and the big constant is due to the data structure we used, not inherent to the algorithm itself.

The tagging procedure is based on heap sort and has cost $O(n\log n)$. Since surface extraction is done locally at octree neighborhoods, each evaluation of the distance field has cost $O(1)$. Thus, the total time complexity of the algorithm is $\Omega(n\log n)$ and its upper bound will depend on the level of refinement of the octree (*i.e.*, on the number of times the implicit function is evaluated during surface extraction).

7 Results

We have tested our algorithms on a series of clean and noisy datasets, including range scans of real objects. Because it is hard to tell the amount of noise present on such range scans, in order to keep control of the amount of noise in each case, we have added noise to originally "clean" models. For each model, the bound to the perturbation added to the position of its original samples is ex-

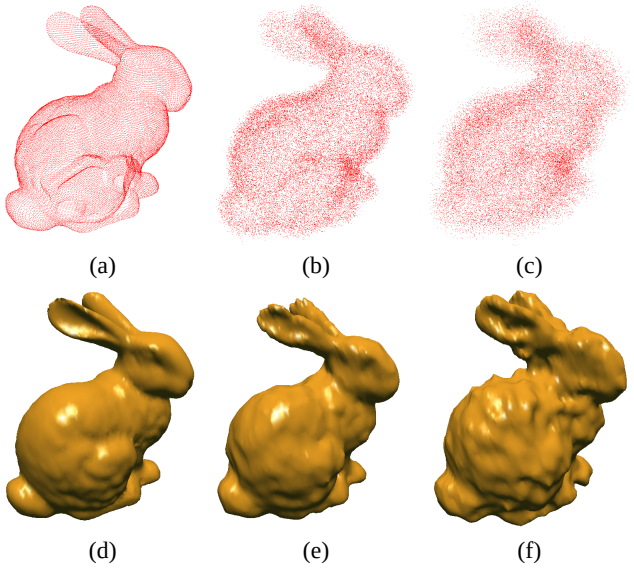

Figure 7: Bunny with noise bounded by (a) 0, (b) 1/60 or (c) 1/30 of the largest side. Reconstructions: (d), (e) or (f), respectively.

pressed as a fraction of the largest size of the bounding box that contains the point cloud (Table 1). No scaling was applied to the input data. Except when explicitly stated otherwise, reconstruction was performed at level 9 of the octree. The relative error bounds shown in Table 1 were obtained by dividing each error bound by the smallest dimension of the corresponding dataset's bounding box. Entries with an asterisk indicate that the input data were obtained from range scans (no precise information about the amount of noisy was available - N/A).

Figures 7 and 10 (left) show several reconstruction results for the Stanford bunny with different levels of noise. Figure 10 (left) also shows the bunny model, this time reconstructed after merging ten range scans containing noise, registration errors and outliers. Note the quality of the reconstructed model. Figure 10 (right) shows the dragon model reconstructed from a set of 60 range scans also containing noise, outliers and registration errors. These examples demonstrate the effectiveness of the proposed technique to handle data captured from the real world.

Figures 1, 6, 8 (a), and 9 (left) show examples of surface reconstruction from clean datasets using the proposed technique. David's head (Figure 1), the dragon (Figure 9(left)) and the buddha (Figure 8(a)) illustrate the case of surface reconstruction involving large datasets. Table 1 provides some statistics associated with the models. The measurements were performed on a Pentium 4 2.2GHz PC with 1GB of memory. The clean models show a bound on the relative error varying from 0.2% to 0.5%. This error varies with the size of the octree leaves and can be further reduced with extra refinement of the octree. For the very noisy example shown in Figure 7(f), the relative error bound is 3.9%.

Although, in practice, it may not be desirable to perform surface reconstruction of very noisy datasets (without previous smoothing), we have included some examples to illustrate the robustness of our approach in the presence of noise (Figures 7(f), 8(c), and 9(right)). The hierarchical representation of oriented charges provides a natural way for creating smoother versions from both clean and noisy datasets. This is illustrated in Figure 8(b), where the Buddha model has been reconstructed using a coarser level of the octree. This is similar to applying a low-pass filter to the geometric information [16]. However, when a very coarse level is used for reconstruction, some dimples may be created (Figure 8(c)), as some mesh vertices may be moved too far by the vertex displacement procedure.

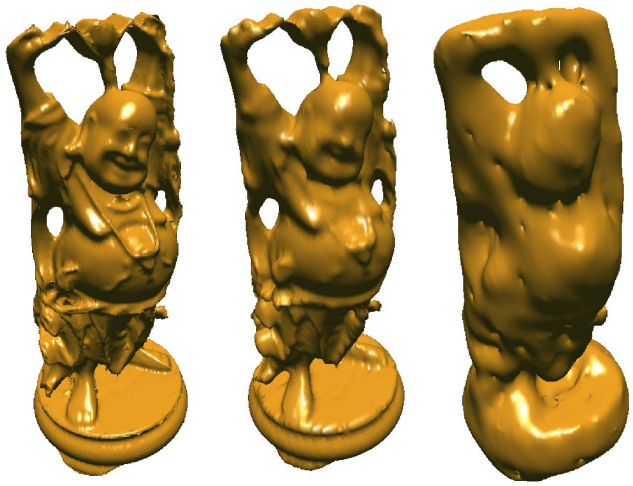

Figure 8: Buddha: (a) No noise (level 9), (b) No noise (level 7), and (c) with noise bounded by 1/50 of the bounding box size (level 9).

We have compared the results of our algorithm with the ones produced by the MPU approach [17]. For noiseless datasets, the visual quality of the reconstructed models are similar, although MPU does a better job in reconstructing sharp features. This situation is illustrated on the left part of Figure 9, where the image at the top was produced using our algorithm and the one at the bottom was created using Ohtake's own code. For the dragon example, MPU's running time was 93 seconds, while our algorithm took 119 seconds.

Our approach is, however, more general in the sense that it handles datasets with minimal information and is considerably more robust to noise. In order to verify this, we added noise bounded by 1% to both the bunny and the dragon point clouds and tried to reconstruct the resulting models using MPU and our approach. Since MPU requires normals, these were computed using Hoppe's normal estimation algorithm [12], considering different numbers of neighbor samples. Figure 11 illustrates the normals obtained using a neighborhood of ten samples for the bunny containing no (left) and 1% noise (right). For these noisy datasets with estimated normals, MPU was unable to reconstruct the models. The RBF-based technique described in [22] was also unable to perform the reconstruction. Figure 9 (top right) shows the reconstructed dragon model produced by our algorithm and illustrates its robustness to the presence of noise. Figure 9 (bottom right) shows the result produced by MPU when the noisy dragon dataset and the normals of the clean model were given as input. For the noisy dragon example, MPU's running time was 466 seconds while our algorithm took only 272 seconds.

7.1 Hole Filling

The implicit function induced by oriented charges provides a natural way for filling small holes in the model. The hierarchical nature of OC's allows for the filling of larger ones. This situation is illustrated in Figure 12. Due to the adaptive subdivision of the octree, nodes inside large holes tend to be at much coarser levels than nodes on the boundary crust. Because the tips of the vectors representing the oriented charges covering such regions will be at the centers of these larger octree nodes, this tends to introduce discrepancies around the boundary of large holes. Therefore, the reconstructed

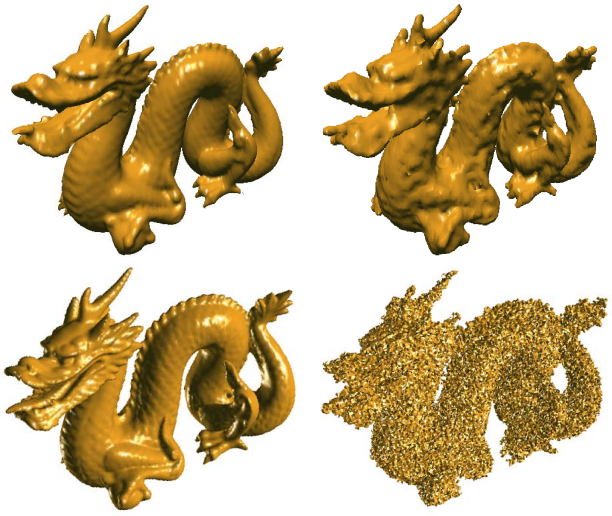

Figure 9: Dragon model reconstructed using our approach (top) and MPU (bottom). Datasets: clean (left) and 1% noise (right).

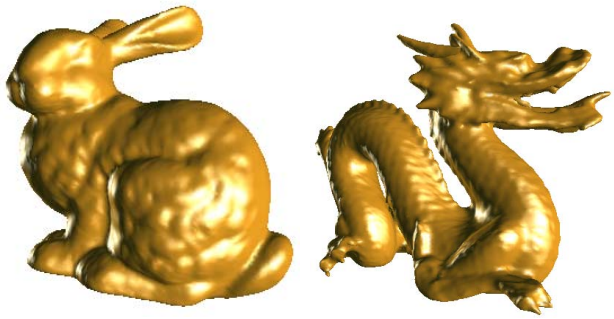

Figure 10: Models reconstructed by our algorithm from actual range scans.

surface for relatively large holes might present some small bumps, as illustrated in Figure 12(right).

8 Conclusions and Future Work

We have introduced a new adaptive and compact framework for representing distance fields efficiently. Our approach can create distance fields using only the position of the input points. As a result, it can be used to reconstruct surfaces from clean as well as noisy datasets. This is an important advantage over previous approaches. Holes are identified and filled by a tagging process in the octree node space, overcoming the common difficulty of using local distance fields. Although our primary focus is surface reconstruction in 3D, the proposed approach seems to naturally generalize to both lower and higher-dimensional spaces.

In order to accelerate iso-surface extraction, one can pre-integrate the blending of the distance fields using fast multipole methods [6] and take advantage of coherence between the oriented charges. To make the hierarchy more efficient, features with greater visual importance need to be analyzed and emphasized (*e.g.*, with more oriented charges dedicated to them) in the hierarchy, resulting in a better adaptive structure.

In our current implementation, the evaluation of the distance field slows down the reconstruction process. This can be improved using either an adaptive polygonizer or a field pre-evaluation approach similar to the fast multipole method [6].

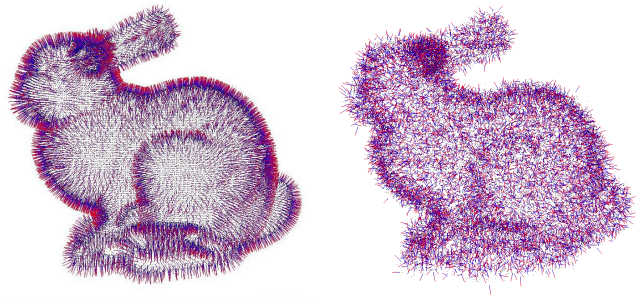

Figure 11: Bunny normals computed from a clean (left) and a 1% noisy point cloud (right).

Table 1: Statistics for the shown examples. Times in seconds and number of points in thousands. Noise and Error bounds.

Model	# of points	Noise	Time OC	Time mesh	Error
Balljoint	137.06	0	8.42	86.80	0.5%
Buddha	543.65	0	12.32	120.32	0.5%
Buddha	543.65	1/50	12.52	123.32	1.5%
Bunny	34.83	0	10.74	90.80	0.5%
Bunny	34.83	1/30	10.66	90.32	3.9%
Bunny*	362.27	N/A	11.28	91.32	N/A
Dragon	437.64	0	11.04	107.22	0.5%
Dragon	437.64	1/100	11.21	108.43	1.5%
Dragon*	1,769.51	N/A	39.28	233.19	N/A
Golf club	209.77	0	8.74	82.80	0.2%
Head	394.18	0	16.32	194.32	0.5%
Vase	68.09	0	7.32	69.77	0.2%

When surface features are smaller than the size of the smallest octree nodes, they may not be appropriately reconstructed. The solution, in this case, is to further subdivide some of the octree cells.

The local nature of oriented charges and of the evaluation process used for reconstruction suggests that the entire procedure (tagging, instantiation of oriented charges and surface extraction) can be performed on the fly for small neighborhoods. This should allow the reconstruction of arbitrarily large models with fine details, using out-of-core techniques.

9 Acknowledgments

We are thankful to the Digital Michelangelo Project, the Stanford 3D Scanning Repository and Cyberware for providing the models used in the paper. This work has been partially supported by grants from NSF CCR-0306438 and FAPERGS (Brazil) Processo No 03/51060.8. Guilherme P. Guimarães implemented the code for

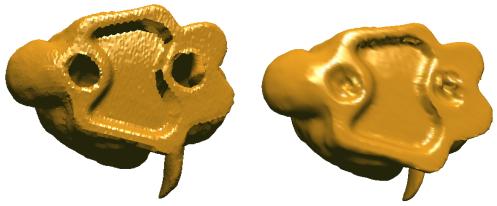

Figure 12: Holes at the bottom of the Bunny (left). Hole filling (right).

estimating the normals from point clouds and the RBF-based approach described in [22].

REFERENCES

[1] B. Adams and P. Dutré. Interactive boolean operations on surfel-bounded solids. *SIGGRAPH*, pages 26–31, 2003.

[2] N. Amenta, M. Bern, and M. Kamvysselis. A new voronoi-based surface reconstruction algorithm. *SIGGRAPH*, pages 415–421, 1998.

[3] F. Bernardini, et al. The ball-pivoting algorithm for surface reconstruction. *IEEE Trans. on Visualization and Computer Graphics*, 5(4):349–359, 1999.

[4] J. Bloomenthal. An implicit surface polygonizer. In Paul Heckbert, editor, *Graphics Gems IV*, pages 324–349. Academic Press, Boston, 1994.

[5] J.C. Carr, R.K. Beatson, B.C. McCallum, W.R. Fright, T.J. McLennan, and T.J. Mitchell. Smooth surface reconstruction from noisy range data. *ACM GRAPHITE*, pages 119–126, 2003.

[6] J.C. Carr, et al. Reconstruction and representation of 3D objects with radial basis functions. *SIGGRAPH*, pages 67–76, 2001.

[7] Digital_Michelangelo. Digital michelangelo project 3d model repository. http://www-graphics.stanford.edu/ projects/mich/.

[8] H.Q. Dinh, G. Turk, and G. Slabaugh. Reconstructing surfaces by volumetric regularization using radial basis functions. *PAMI*, 24(10):1358–1371, 2002.

[9] H. Edelsbrunner and E.P. Muche. Three-dimensional alpha shapes. *ACM TOG*, 13:43–72, 1994.

[10] R. Franke. Scattered data interpolation: tests of some methods. *Mathematics of Computation*, 38:181–200, 1982.

[11] R. Franke and G. Nielson. Smooth interpolation of large sets of scattered data. *Int. Journal for Numerical Methods in Engeneering*, 15:1691–1704, 1980.

[12] H. Hoppe, et al. Surface reconstruction from unorganized points. *SIGGRAPH*, pages 71–78, 1992.

[13] W. Jeong, et al. Automatic generation of subdivision surface head models from point cloud data. *Graphics Interface'02*, pages 181–188, 2002.

[14] B. Morse, T. Yoo, P. Rheingans, D. Chen, and K. Subramanian. Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions. *Shape Modeling International*, pages 89–98, 2001.

[15] L. Nyland, et al. The impact of dense range data on computer graphics. *Multi-View Modeling and Analysis Workshop*, pages 3–10, 1999.

[16] Y. Ohtake, A. Belyaev, and H. Seidel. A multi-scale approach to 3D scattered data interpolation with compactly supported basis functions. *Shape Modeling International'03*, 2003.

[17] Y. Ohtake, et al. Multi-level partition of unity implicits. *SIGGRAPH*, pages 463–470, 2003.

[18] H. Samet. *The design and analysis of spatial data structures*. Addison-Wesley Longman Publishing Co., Inc., 1990.

[19] S. Schaefer and J. Warren. Dual marching cubes: Primal contouring of dual grids. *Pacific Graphics*, pages 70–76, 2004.

[20] R. Szeliski and D. Tonnesen. Surface modeling with oriented particle systems. *SIGGRAPH'92*, 26:185–194, 1992.

[21] D. Terzopoulos and D. Metaxas. Dynamic 3d models with local and global deformations: Deformable superquadrics. *PAMI*, 13:703–714, 1991.

[22] I. Tobor, P. Reuter, and C. Schlick. Efficient reconstruction of large scattered geometric datasets using the partition of unity and radial basis functions. *Journal of WSCG*, 12:467–474, 2004.

[23] G. Turk and J. O'Brien. Variational implicit surfaces. Technical Report GIT-GVU-99-15, Georgia Institute of Technology, 1999.

[24] H. Wendland. Piecewise polynomial, positive definite and compactly supported radial basis functions of minimum degree. *AICM*, 4:389–396, 1995.

[25] H.K. Zhao, S. Osher, and R. Fedkiw. Fast surface reconstruction using the level set method. *First IEEE Workshop on Variational and Level Set Methods*, pages 194–202, 2001.